Differential Fault Analysis

Marc Joye* and Michael Tunstall†

Synonyms

Fault analysis, Fault attacks, Fault injection; Collision fault analysis.

Definitions

Differential fault analysis is an active attack against cryptographic implementations. The goal is to induce faults during a cryptographic operation to infer private information (e.g., a decryption key).

Background

Cryptographic systems should not only be resistant to cryptanalysis, they should also be resistant to implementation attacks, including side-channel and fault attacks. *Differential fault analysis* was developed by Boneh et al (2001) and extended to the symmetric-key setting by Biham and Shamir (1997). The principle idea behind fault attacks consists in modifying the normal behavior of a cryptographic implementation in order to get a faulty output. Then from one or more faulty outputs, the attacker tries to infer some information about the secret key. Examples of practical fault injection are described in Skorobogatov and Anderson (2002); Bar-El et al (2006). When successful, those attacks are very powerful; for example, applied to the AES, only one pair of correct/faulty ciphertexts suffice to recover the whole secret key. Of course, numerous countermeasures have been proposed to avoid, prevent, or detect fault attacks.

* Corresponding author
Zama, Paris, France
marc@zama.ai

† Rambus Cryptography Research, San Francisco, CA, USA
michael.tunstall@cryptography.com

We refer the reader to Joye and Tunstall (2012) for a detailed account on various aspects of fault analysis, including countermeasures.

Analyzing Block Ciphers

The first example of a fault attack applied to a block cipher was described by Biham and Shamir (1997), who noted that injecting a fault could be analyzed using techniques from differential cryptanalysis.

Differential cryptanalysis operates by assuming that an internal state in two instances of a block has a small difference. That is, for two instances of the same block cipher they have nearly the same state after a certain number of rounds of the block cipher and have some small difference. An attacker would then determine how this difference would propagate through the remaining rounds of a block cipher. This small difference will typically impose some structure in the `XOR` difference between the two instances in subsequent rounds of the block cipher.

If one is able to inject a fault into a block cipher one can compare a normal instance with a faulty one, where a small error has been injected while encrypting the same plaintext. This is a significant advantage over traditional differential cryptanalysis, where one would have to wait for the correct circumstances to occur by encrypting a large number of plaintexts. When faults are used to cause a small error followed by differential cryptanalysis the attack is typically referred to as differential fault analysis.

When applied to AES, it has been shown that a single fault is sufficient to derive a 128-bit AES key (Tunstall et al 2011). If one byte is modified in the input to the penultimate `MixColumns` operation, then a four byte difference will occur in the input to the last `MixColumns` operation. The difference on the output of the `MixColumns` operations can be computed based on the possible values for the four-byte difference. One can then construct a set of equations involving the last sub-key and conduct four 32-bit search for sections of the last subkey that satisfy these equations.

For example, given a fault in the input to the eighth round, considering the state of the differences after the ninth round shift row, one can obtain the following set of equations that include the values of the last subkey key bytes k_1, k_8, k_{11} and k_{14}, thus giving an expression for 32 key bits:

$$\begin{aligned} 2\,\delta &= S^{-1}(x_1 \oplus k_1) \oplus S^{-1}(x'_1 \oplus k_1) \\ \delta &= S^{-1}(x_{14} \oplus k_{14}) \oplus S^{-1}(x'_{14} \oplus k_{14}) \\ \delta &= S^{-1}(x_{11} \oplus k_{11}) \oplus S^{-1}(x'_{11} \oplus k_{11}) \\ 3\,\delta &= S^{-1}(x_8 \oplus k_8) \oplus S^{-1}(x'_8 \oplus k_8) \end{aligned}$$

where k_1, k_8, k_{11} and k_{14} are all unknown values in $\{0,\ldots,255\}$, δ is the effect of the fault propagated to the ninth round also in $\{0,\ldots,255\}$, x_1, x_{14}, x_{11}, x_8 are ciphertext bytes from the correct ciphertext, x'_1, x'_{14}, x'_{11}, x'_8 are ciphertext bytes from the faulty ciphertext and S^{-1} is the inverse of the `SubBytes` function.

Using the above an attacker would select a value for δ and determine which values of k_1, k_8, k_{11} and k_{14} satisfy the equations using four independent exhaustive searches. Each equation will return 0, 2, or 4 hypotheses (Nyberg 1993) and in total should return 2^8 hypotheses for the 32 bits of the last

subkey. This can be repeated with another three sets of equations for the remaining bits of the last subkey, resulting in an exhaustive search of 2^{32} to determine a 128-bit AES secret key.

Another example of differential fault analysis is to inject a fault in the early rounds of a block cipher. One can then do an exhaustive search for the plaintext that produces the incorrect ciphertext when no fault is injected. The search space is dictated by whatever hypotheses an attacker can make about the effect of the injected fault. For example, affecting one chosen byte in the second round will give an exhaustive search of size 2^{32}. The attack then proceeds with a similar strategy to that described above, but comparing plaintexts rather than ciphertexts. This is typically referred to as collision fault analysis (Blömer and Seifert 2003; Hemme 2004).

Analyzing Public-Key Algorithms

Differential fault analysis is not limited to block ciphers. Public-key cryptographic algorithms typically make use of hard mathematical problems to provide security. Hence, attacks and countermeasures can make use of the algebraic properties of the cryptographic algorithms. This is illustrated below, showing how faults can be applied to RSA signatures; see Boneh et al (2001).

Let $N = pq$ be the product of two large (equal-size) distinct prime integers. Let also a public exponent e co-prime to Euler's function $\varphi(N) = (p-1)(q-1)$ and corresponding private exponent $d = 1/e \bmod \varphi(N)$. The public key is $\{N, e\}$ while the private key is d. Imagine that the signature of a message m is computed as $\sigma \leftarrow \mu(m)^d \bmod N$ for some deterministic padding function μ (e.g., Full-Domain Hash or FDH). Using public key $\{N, e\}$, anyone can verify the validity of signature σ on message m by checking that $\sigma^e \bmod N \stackrel{?}{=} \mu(m)$.

RSA signing can be sped up using Chinese remaindering (a.k.a. CRT mode). In this case, the private key is given by $\{p, q, d_p, d_q, i_q\}$ with $d_p = d \bmod (p-1)$, $d_q = d \bmod (q-1)$, and $i_q = q^{-1} \bmod p$. Signature σ on message m is generated by evaluating two half exponentiations, $\sigma_p \leftarrow \mu(m)^{d_p} \bmod p$ and $\sigma_q \leftarrow \mu(m)^{d_q} \bmod q$, and then $\sigma \leftarrow \mathrm{CRT}(\sigma_p, \sigma_q) := \sigma_q + q\big(i_q(\sigma_p - \sigma_q) \bmod p\big)$. This yields an expected speed-up factor of 4.

Now suppose that one of the two half exponentiations is faulty; say, suppose one gets the faulty signature $\sigma' = \mathrm{CRT}(\sigma'_p, \sigma_q)$ where $\sigma'_p \neq \sigma_p$ and σ_q is correct. It is easily seen that a simple GCD (greatest common divisor) computation enables the recovery of private parameter q as $q = \gcd(\sigma' - \sigma \pmod N, N)$, and next of $p = N/q$. Actually, the faulty signature suffices to recover p and q as $\gcd(\sigma'^e - \mu(m) \pmod N, N) = q$.

RSA can also be used for encryption by "exchanging" the roles of e and d. The encryption of a message m is given by $C \leftarrow \mu(m)^e \bmod N$ for some padding function μ. For security reasons, the padding used for encryption must be probabilistic, such as OAEP. The previous attack does not apply against e.g. RSA-OAEP since the correctness of the plaintext message m is explicitly checked by the decryption algorithm. Note also that the attacker has no idea of the random coins used in the evaluation

of $\mu(m)$—remember that μ is always probabilistic for RSA encryption.

Countermeasures

Countermeasures to fault attacks typically rely on adding redundancy. This can vary from low-level countermeasure, such as adding hardware checksums to registers, to high-level countermeasures, such as repeating algorithms and verifying that the operations produce the same output. Alternatively, one can rely on so-called infective computation (Yen et al 2003) where an error results in a faulty output that leaks no information.

In the RSA example above one can verify an RSA signature with a very small decrease in performance. A public key exponent is typically set to $2^{16}+1$ meaning that a verification requires 17 multiplications. However, it has been shown that one could inject two faults, one in the RSA and the second where the verification is evaluated (Kim and Quisquater 2007).

Typically, it may only necessary to consider single faults for a single cryptographic operation, on condition that faults can be detected and cryptographic keys can be destroyed or made inaccessible. That is, if an operating system repeatedly sees a status indicating that a fault is being injected a response should occur. Depending on the application, this response could include zeroing cryptographic keys or deactivating a device. An attacker will need to characterize the effect of single faults before attempting more complex attacks. If an attacker can make many attempts to inject faults then it may be possible to overcome complex countermeasures. That is, if a countermeasure is designed to prevent n faults then it may be overcome with $n+1$ faults.

When defining countermeasures to fault attacks it is important to define a model of what an attacker is capable of and design appropriate countermeasures. Without such a model, there is always one more attack possible with one more fault. The second part is to define the response to detecting a fault attack. If a device would permanently deactivate then the fault countermeasures could be quite lightweight and, conversely, if no response is made then the fault countermeasures would need to be very robust.

Cross-References

Fault attack, Electromagnetic fault injection, Fault sensitivity analysis, Ineffective fault attack, Statistical fault attack, Evolution of fault attacks on cryptosystems.

References

Bar-El H, Choukri H, Naccache D, Tunstall M, Whelan C (2006) The sorcerer's apprentice guide to fault attacks. Proc IEEE 94(2):370–382, doi:10.1109/JPROC.2005.862424

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski Jr BS (ed) Advances in Cryptology – CRYPTO '97, Springer, Lecture Notes in Computer Science, vol 1294, pp 513–525, doi:10.1007/BFb0052259

Blömer J, Seifert J (2003) Fault based cryptanalysis of the advanced encryption standard (AES). In: Wright RN (ed) Financial Cryptography (FC 2003), Springer, Lecture Notes in Computer Science, vol 2742,

pp 162–181, doi:10.1007/978-3-540-45126-6_12

Boneh D, DeMillo RA, Lipton RJ (2001) On the importance of eliminating errors in cryptographic computations. J Cryptol 14(2):101–119, doi:10.1007/s001450010016, Earlier version appeared in Proc. of EUROCRYPT '97

Hemme L (2004) A differential fault attack against early rounds of (triple-)DES. In: Joye M, Quisquater JJ (eds) Cryptographic Hardware and Embedded Systems – CHES 2004, Springer, Lecture Notes in Computer Science, vol 3156, pp 254–267, doi:10.1007/978-3-540-28632-5_19

Joye M, Tunstall M (eds) (2012) Fault Analysis in Cryptography. Information Security and Cryptography, Springer, doi:10.1007/978-3-642-29656-7

Kim CH, Quisquater JJ (2007) Fault attacks for CRT based RSA: new attacks, new results, and new countermeasures. In: Sauveron D, et al (eds) Information Security Theory and Practices (WISTP 2007), Springer, Lecture Notes in Computer Science, vol 4462, pp 215–228, doi:10.1007/978-3-540-72354-7_18

Nyberg K (1993) Differentially uniform mappings for cryptography. In: Helleseth T (ed) Advances in Cryptology – EUROCRYPT '93, Springer, Lecture Notes in Computer Science, vol 765, pp 55–64, doi:10.1007/3-540-48285-7_6

Skorobogatov SP, Anderson RJ (2002) Optical fault induction attacks. In: Kaliski Jr BS, et al (eds) Cryptographic Hardware and Embedded Systems – CHES 2002, Springer, Lecture Notes in Computer Science, vol 2523, pp 2–12, doi:10.1007/3-540-36400-5_2

Tunstall M, Mukhopadhyay D, Ali S (2011) Differential fault analysis of the advanced encryption standard using a single fault. In: Ardagna CA, Zhou J (eds) Information Security Theory and Practice (WISTP 2011), Springer, Lecture Notes in Computer Science, vol 6633, pp 224–233, doi:10.1007/978-3-642-21040-2_15

Yen SM, Kim S, Lim S, Moon SJ (2003) RSA speedup with Chinese remainder theorem immune against hardware fault cryptanalysis. IEEE Transactions on Computers 52(4):461–472, doi:10.1109/TC.2003.1190587